SAT: Designing for Testability

1. Three developers could benefit from improving either the observability or the controllability of the system/class they are testing, but each developer encounters a problem. State whether each of the problems relates to observability or controllability.

a) Developer 1: “I can’t assert whether the method under test worked well.”

b) Developer 2: “I need to make sure this class starts with a boolean set to false, but I can’t do it.”

c) Developer 3: “I instantiated the mock object, but there’s no way to inject it into the class.”

2. Sarah has joined a mobile app team that has been trying to write automated tests for a while. The team wants to write unit tests for part of their code, but they tell Sarah, “It’s hard.” After some code review, the developers list the following problems in their code base:

a) Many classes mix infrastructure and business rules.

b) The database has large tables and no indexes.

c) There are lots of calls to libraries and external APIs.

d) Some classes have too many attributes/fields.

To increase testability, the team has a budget to work on two of these four issues. Which items should Sarah recommend that they tackle first?
Note: All four issues should be fixed, but try to prioritize the two most important ones. Which influences testability the most?

3. Think about your current project. Are parts of it hard to test? Can you explain why? What can you do to make it more testable?

4. How can you improve the testability of the following OrderDeliveryBatch class?

```java
public class OrderDeliveryBatch {
    public void runBatch() {
        OrderDao dao = new OrderDao();
        DeliveryStartProcess delivery = new DeliveryStartProcess();
        List<Order> orders = dao.paidButNotDelivered();
        for (Order order : orders) {
            delivery.start(order);
            if (order.isInternational()) {
                order.setDeliveryDate("5 days from now");
            } else {
                order.setDeliveryDate("2 days from now");
            }
        }
    }
    class OrderDao {
        // accesses a database
        class DeliveryStartProcess {
            // communicates with a third-party web service
        }
    }
}
```

5. How can you improve the testability of the following KingsDayDiscount class?

```java
public class KingsDayDiscount {
    public double discount(double value) {
        Calendar today = Calendar.getInstance();
        boolean isKingsDay = today.get(MONTH) == Calendar.APRIL
                && today.get(DAY_OF_MONTH) == 27;
        return isKingsDay ? value * 0.15 : 0;
    }
}
```